Romain Jacquet **BTS SIO 1**
17/12/2025

Compte rendu TP Atelier 13 : Javascript

Sommaire

Introduction

Lors de l'Atelier 13, j'ai travaillé sur le JavaScript, surtout sur la manipulation du DOM et la gestion des événements. Cette séance, le 17 décembre 2025, m'a permis de voir comment rendre une page web interactive. Grâce à plusieurs exercices, j'ai appris à gérer des clics de souris et même des raccourcis clavier pour rendre les pages plus dynamiques.

Navigateur 3

Cacher sur clic

Ajouter JavaScript à la `button` pour faire `<div id="text">` disparaître quand on clique dessus.

Se cacher

Crée un bouton qui se cache tout seul au clic.

```
<div id="text">Ceci est le texte qui va disparaître.</div>

<button id="bouton-exo1">Cacher le texte</button>
```

J'ai donner un id au bouton pour que le fichier .js puisse le trouver

```
var btn1 = document.getElementById("bouton-exo1");
btn1.addEventListener("click", function() {
    document.getElementById("text").style.display = "none";
});
```

On récupère le bouton est quand un clic est fait sur le bouton on cache le texte :

Ceci est le texte qui va disparaître.

Cacher le texte

Cacher le texte

Quels handlers exécutent?

Il y a un bouton dans la variable. Il n'y a aucun : handlers dessus.

Quels handlers s'exécutent au clic après le code suivant? Quelles alertes apparaissent?

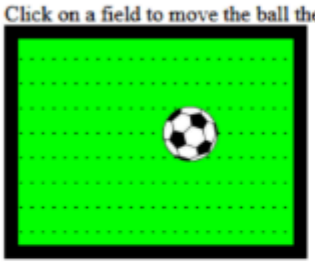
```
1  button.addEventListener("click", () => alert("1"));
2
3  button.removeEventListener("click", () => alert("1"));
4
5  button.onclick = () => alert(2);
```

Pour cette exercice les deux alerte vont apparaître “1” et “2”

Déplacez le ballon à travers le terrain

Déplacez le ballon à travers le terrain jusqu'à un clic.

ballon2.html



Exigences:

- Le centre de la balle devrait passer exactement sous le pointeur au clic (si possible sans franchir le bord du champ).
- L'animation CSS est la bienvenue.
- La balle ne doit pas franchir les limites du terrain.
- Quand la page est défilée, rien ne devrait se casser.

Notes:

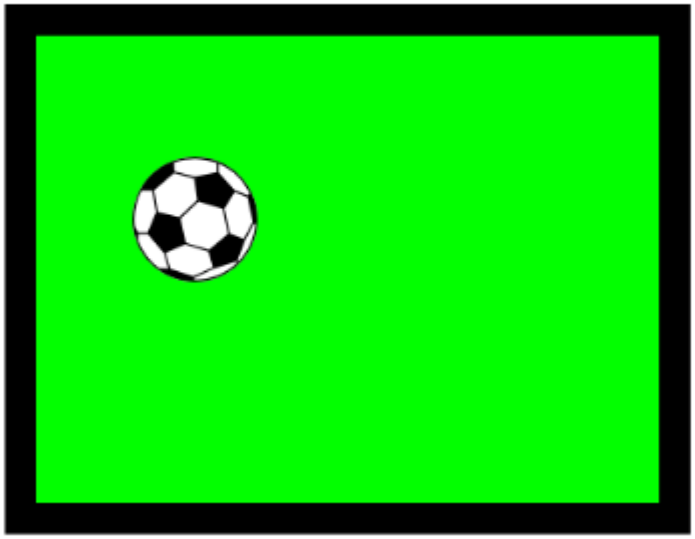
- Le code doit aussi fonctionner avec différentes tailles de balle et de champ, sans être lié à des valeurs fixes.
- Propriétés d'utilisation `event.clientX/event.clientY` pour les coordonnées de clic.

j’ai utilisé le code ci-dessous

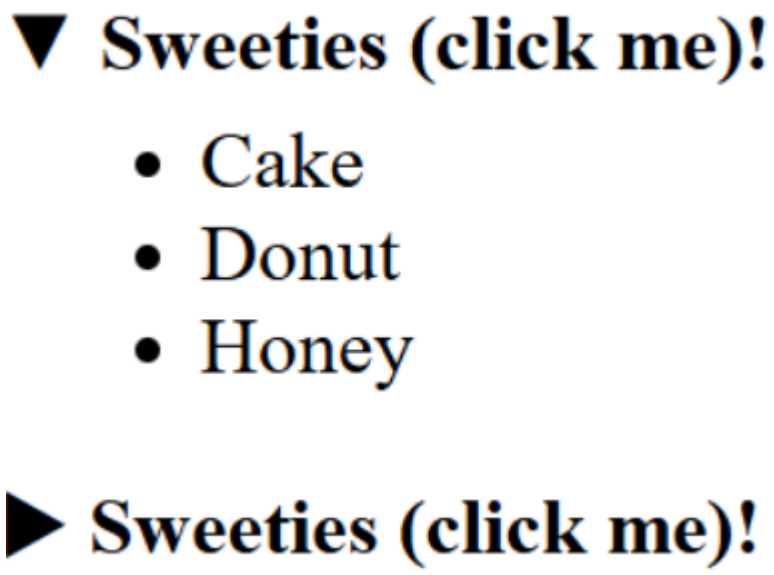
```
let field = document.getElementById("field");
let ball = document.getElementById("ball");

field.onclick = function(event) {
  let fieldCoords = this.getBoundingClientRect();
  let ballTop = event.clientY - fieldCoords.top - field.clientTop - (ball.clientHeight / 2);
  let ballLeft = event.clientX - fieldCoords.left - field.clientLeft - (ball.clientWidth / 2);

  if (ballTop < 0) ballTop = 0;
  if (ballLeft < 0) ballLeft = 0;


  if (ballLeft + ball.clientWidth > field.clientWidth) {
    ballLeft = field.clientWidth - ball.clientWidth;
  }
  if (ballTop + ball.clientHeight > field.clientHeight) {
    ballTop = field.clientHeight - ball.clientHeight;
  }

  ball.style.left = ballLeft + "px";
  ball.style.top = ballTop + "px";
```

Voici le rendu

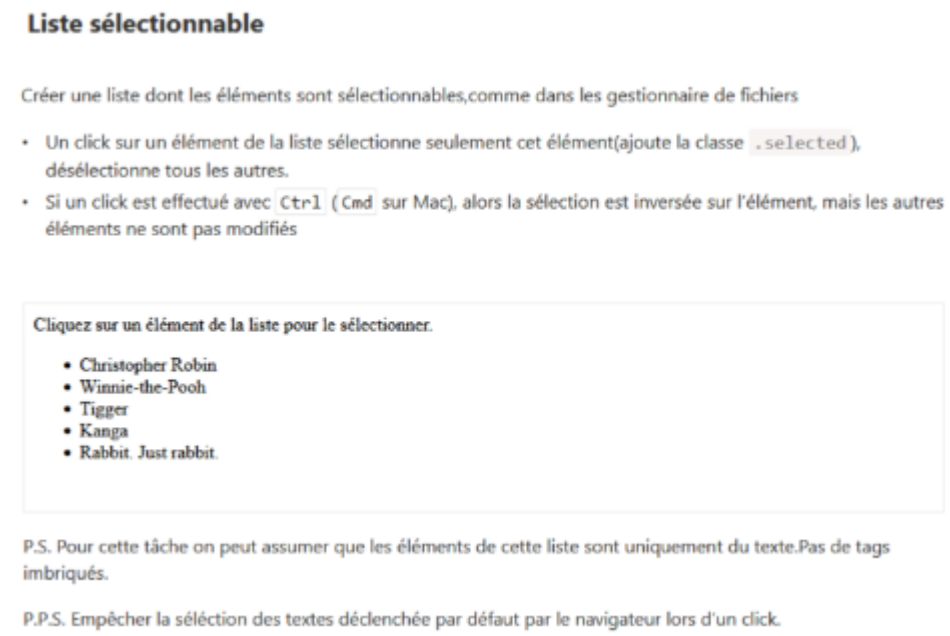
Click on a field to move the ball there.
The ball should never leave the field.



```
<script>
  let title = document.getElementById('title');
  let list = document.getElementById('list');

  title.addEventListener('click', function() {

    let isHidden = (list.style.display === 'none');

    if (isHidden) {
      list.style.display = 'block';
      title.innerHTML = '▼ Sweeties (click me)!';
    } else {
      list.style.display = 'none';
      title.innerHTML = '▶ Sweeties (click me)!';
    }
  });
</script>
```



Navigateur 4



Pour cette exercice j’ai utilisé le code ci-dessous

```
const ul = document.getElementById('ul');

ul.addEventListener('click', function(event) {

  if (event.target.tagName !== "LI") return;

  if (event.ctrlKey || event.metaKey) {
    event.target.classList.toggle('selected');
  } else {

    let selectedItems = ul.querySelectorAll('.selected');
    for (let elem of selectedItems) {
      elem.classList.remove('selected');
    }
    event.target.classList.add('selected');
  }
});

ul.onmousedown = function() {
  return false;
};
```

Ce qui donne le résultat ci-dessous !!!!

Cliquez sur un élément de la liste pour le sélectionner.

- Christopher Robin
- Winnie-the-Pooh
- Tigger
- Kanga
- Rabbit. Just rabbit.

Raccourcis clavier étendus

Créer une fonction `runOnKeys(func, code1, code2, ... code_n)` exécutant la fonction `func` lorsqu'on appuie simultanément sur les touches avec les codes suivant `code1`, `code2`, ..., `code_n`.

Par exemple, le code ci-dessous montre `alert` lorsque `"Q"` et `"W"` sont appuyées ensemble (dans n'importe quelle langue, avec ou sans l'activation de La touche Majuscule, CapsLock)

```
runOnKeys(
  () => alert("Hello!"),
  "KeyQ",
  "KeyW"
);
```

ide :

Nous devons utiliser deux gestionnaires: `document.onkeydown` et `document.onkeyup`.

L'ensemble `pressed` doit garder les touches en cours appuyées.

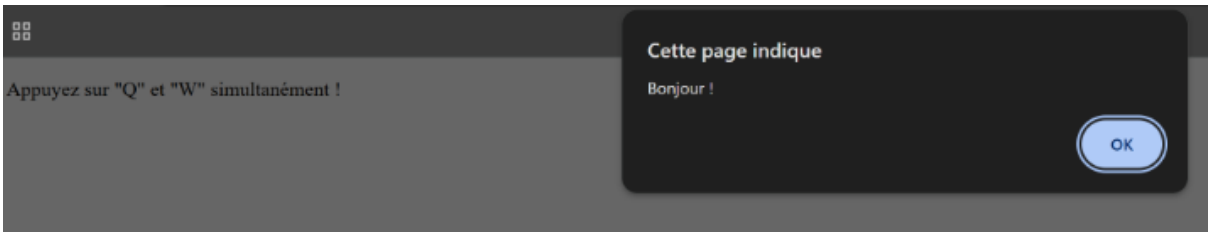
Créons un set `pressed = new Set()` pour garder les touches actuellement enfoncées.

Le premier gestionnaire en ajoute, tandis que le second en supprime. Chaque fois sur `keydown` nous vérifions si nous avons suffisamment de touches enfoncées et exécutons la fonction si c'est le cas.

Pour cette exercice j'ai utilisé le code ci-dessous

```
function runOnKeys(func, ...codes) {
  let pressed = new Set();

  document.addEventListener('keydown', function(event) {
    pressed.add(event.code);

    console.log("Touche détectée : " + event.code);

    let allConnect = true;
    for (let code of codes) {
      if (!pressed.has(code)) {
        allConnect = false;
        break;
      }
    }

    if (allConnect) {
      pressed.clear();
      func();
    }
  });

  document.addEventListener('keyup', function(event) {
    pressed.delete(event.code);
  });
}

runOnKeys(
  () => alert("Combinaison réussie !"),
  "KeyQ",
  "KeyW"
);
```

Voici le résultat final !!!!



Conclusion

Pour conclure, cet atelier m’a permis de mettre en pratique le JavaScript sur des exemples concrets. J’ai commencé par des actions simples, comme cacher des éléments au clic, puis j’ai appris :

- à animer des objets selon la position de la souris,
- à gérer plusieurs sélections dans une liste avec la touche Ctrl,
- à déclencher des actions avec des combinaisons de touches au clavier.

Ces exercices m’ont aidé à mieux comprendre comment JavaScript fonctionne avec le HTML et le CSS pour rendre une page web interactive.